HARD COPY RENDITION OF ODA-DOCUMENTS

J. BEHRMANN-POITIERS, H. KEIL, H. LOEBL

Philips Research Laboratory Hamburg

ABSTRACT

The standardization of the Office Document Architecture (ODA) of ISO DIS 8613 covers at present documents comprising text, raster graphics, and geometric graphics. Furthermore, ISO DIS 8613 defines the Office Document Interchange Format (ODIF) for the interchange of office documents between different host systems.

This paper describes the imaging process, which renders a document in a human perceptible form. It converts a document from ODIF into a printer dependent bitmap via a suitable internal document representation, which is based on the formal data definitions of the ODA standard. Such a conversion should be performed preferably in the printer controller.

In addition, a proposal will be discussed to extend the raster graphics content architecture by grey-scale and colour images. For this purpose the CEPT Videotex standard seems to be a suitable base.

The above mentioned solution for the rendition of ODA documents will be compared to an approach using PostScript, a procedural page description language, where the same internal document representation is converted to the corresponding PostScript format, which is subsequently transmitted to a PostScript printer.

1. Introduction

Desktop Publishing [1] is a relatively new field of electronic document processing. It is associated with the generation and rendition of complex and high quality documents. Unfortunately, it is seldom possible to exchange documents in electronic form between different document processing systems. The reason is that each processing system uses its own specific and therefore incompatible document architecture.

In order to solve this problem, ISO currently standardizes a so-called "Office Document Architecture" (ODA), which is defined in ISO DIS 8613 [2]. ODA is a collection of rules that allow the hierarchical composition of a document, either in logical or in layout form. Content types like character text, raster graphics, and geometric graphics are specified in so-called content architectures.

Besides a document architecture, ISO 8613 also defines an "Office Document Interchange Format" (ODIF) [3] which supports the electronic interchange of

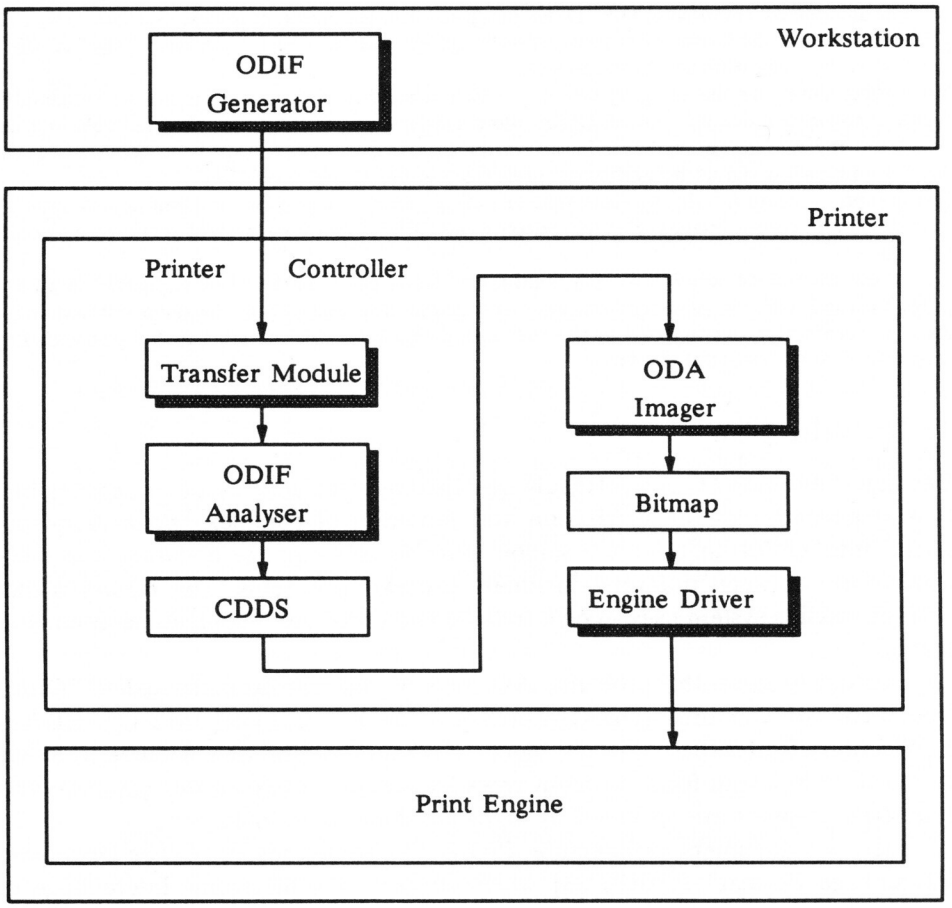
documents between different host systems. When documents are interchanged in ODIF form, three modes can be distinguished: the processable form, the formatted form and the formatted processable form. The processable form contains a logical document structure and content, which can be further edited by the receiver. The formatted form contains the layout document structure and content and can only be displayed or printed. Finally, the formatted processable form contains both the logical and the layout structure.

The various steps necessary to convert an ODIF document from its formatted form into a bitmap for hard copy rendition will be discussed in this paper.

General Aspects

The block diagram of figure 1 illustrates the flow of data from the host system to the printing engine. An ODIF byte–stream is analysed by the ODIF–Analyser,

Figure 1: Conversion of an ODIF Byte–Stream into a Bitmap



which generates a hierarchical tree structure, the so–called Common Document Data Structure (CDDS) and calls the rasterization process, also termed ODA–Imager. This process performs a transformation into a bitmap suitable for the rendition on a page printer. In order to relieve the host system from a time consuming task, the whole ODIF–to–bitmap conversion will be performed by the printer controller.

According to ISO 8613 part 5 data stream B is associated with the formatted form of an ODIF file and consists of one document profile descriptor and may contain the following interchange data units:

- layout object descriptor
- layout class descriptor
- presentation style descriptor
- text unit

A layout object descriptor contains all the information, i.e. attributes and references to subordinate objects, which define the layout of the corresponding specific object (page, frame, block) and its relationship to other specific layout objects. A layout class descriptor describes a generic layout object. A presentation style descriptor is a collection of attributes that define the presentation of a content portion within a basic object, i.e. a block. The content portions itself, like character text or graphics, are included in text units.

ODIF–Analyser

Because the ODIF form of a document is not well suited for its further processing, it is at first converted into an internal document representation by means of the ODIF–Analyser. The ODIF–Analyser is created by a generator program, which in turn is generated by means of the Unix software tools LEX and YACC by using the formal data definitions of ODIF in ISO DIS 8613 part 5. Since the standardization of ODA/ODIF is not yet completed and changes must be expected in the future, this approach has the advantage that a fast adaptation of the ODIF–Analyser to the latest version of ISO 8613 is feasible. A discussion of this generator program is outside the scope of this paper.

The internal representation of a document created by the ODIF–Analyser is a hierarchical tree structure. It is based on a so–called Common Document Data Structure (CDDS) which is implemented in the programming language C and was automatically derived by a suitable generator program from the formal data definitions of ISO 8613.

Figure 2 illustrates the tree structure of a layout structured document with composite objects like document as the root of the tree, page and frame. Blocks at the lowest level are basic objects of this tree. Content portions constitute the leaves and are connected to block nodes. All the various attributes can be found in the tree nodes and the associated data structures.

Figure 2: Data Structure (CDDS) of a Layout Structured Document

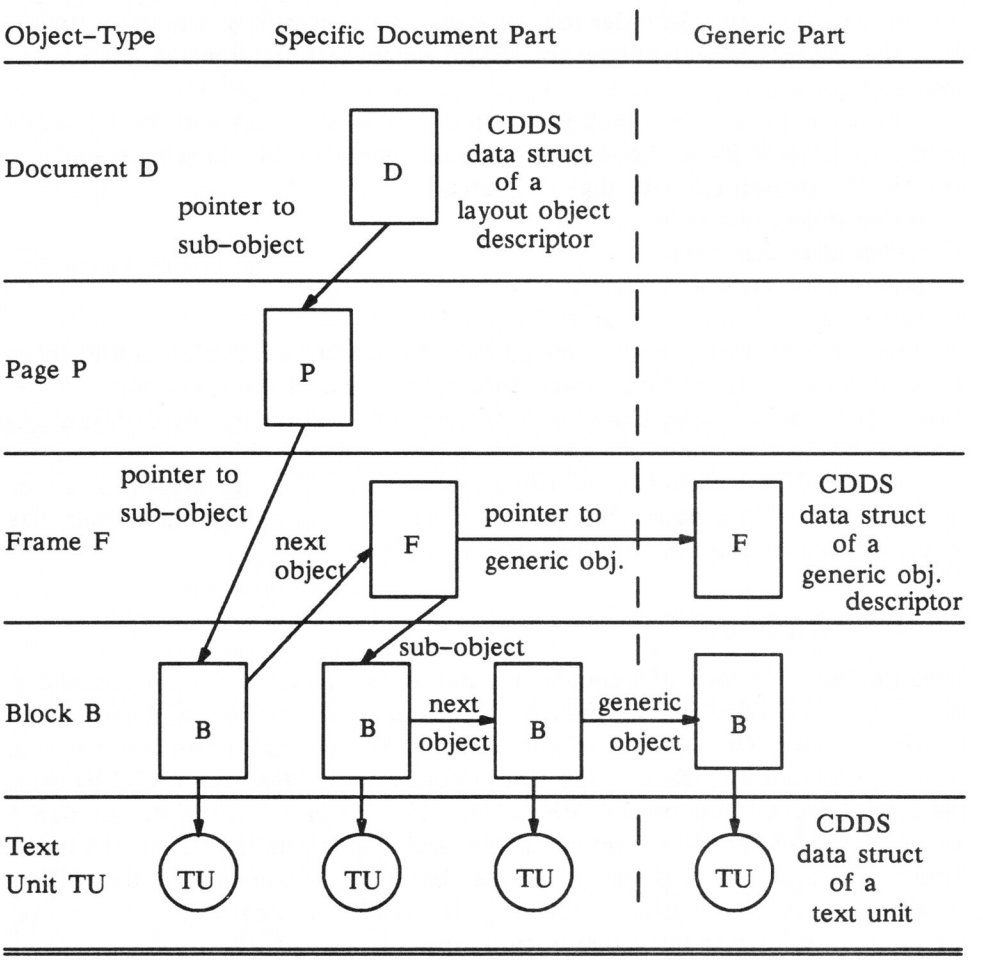


Figure 3 and figure 4 present the formal data definition of a layout object descriptor and its representation as a CDDS struct. The advantage of such an internal representation results from the fact that the access to the attributes (variables) and the subordinate or peer layout objects, generic objects and presentation styles is very convenient.

Figure 3: Formal Definition of a Layout-Object-Descriptor according to ISO 8613/5

```
Layout-Object-Descriptor        ::= SEQUENCE {
    object-type                 Layout-Object-Type OPTIONAL,
    descriptor-body             Layout-Object-Descriptor-Body
                                        OPTIONAL }

Layout-Object-Type              ::= INTEGER {
                                document-layout-root (0), page-set (1),
                                page (2), frame (3), block (4) }

Layout-Object-Descriptor-Body   ::= SET {
    object-identifier           Object-or-Class-Identifier OPTIONAL,
    subordinates                [0] IMPLICIT SEQUENCE OF
                                        Numeric String OPTIONAL,
    . . .
    position                    [3] IMPLICIT Measure-Pair OPTIONAL,
    dimensions                  [4] IMPLICIT Measure-Pair OPTIONAL,
    . . .}
```

Figure 4: C-Data Representation of a layout-Object_Descriptor

```
struct layout_Object_Descriptor {           /* SEQUENCE */
    INTEGER                         object_type;
    Layout_Object_Descriptor_Body*  descriptor_body; };

struct layout_Object_Descriptor_Body {      /* SET */
    PrintableString                 object_identifier;
    SEQUENCEOF_NumericString        subordinates;
    . . .
    Measure_Pair*                   position;
    Measure_Pair*                   dimensions;
    . . .};
```

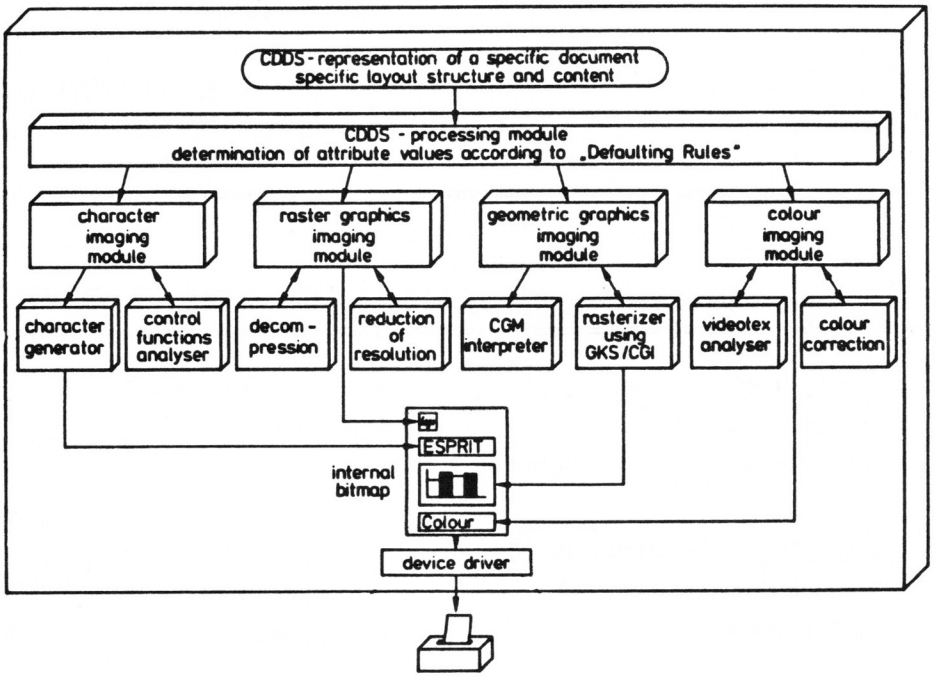
ODA–Imager

In the document processing model of ISO 8613, the following three processes are mentioned:

- the editing process : it handles the logical structure and content of a document
- the formatting process : it generates the layout structure of a document
- the imaging process : it renders the document in a human–perceptible form on a presentation medium.

The ODA–Imager corresponds to the imaging process of this model. It processes the specific layout structure of a document, the associated content portions, generic layout parts, and presentation styles and generates a format suitable for printing / displaying. Figure 5 presents a module structure of the ODA–Imager.

Figure 5: Module Structure of the ODA–Imager



This module structure can be subdivided into the following parts:
- a layout structure processing part,
- separate content imaging parts,
- a device–adaptation part.

The main task of the layout structure processing module is the determination of attribute values by means of the imaging rules for documents as defined in ISO 8613. A central role plays the determination of attribute values, such as position, dimensions, and presentation attributes according to ODA's "Defaulting Rules". The value of an attribute may be specified in different parts of a document's layout structure:
- in the specific layout object directly
- in a presentation style, which is referenced by a pointer from the specific object
- in a generic layout object which is referenced from the specific object
- in a presentation style, which is referenced from a generic layout object, which in turn is referred to from the specific layout object
- in a default value list specified at a higher level of the hierarchical layout structure
- the default value defined by ISO 8613

An attribute takes the value delivered by the first of these rules which is applicable.

Further rules guide the positioning and dimensioning of the layout objects such as frames and blocks within a page. The origin of the orthogonal page coordinate system is the top left corner on the page. Position and dimensions are specified in "Basic Measurement Units" (BMU). Frames and blocks are positioned relative to the layout object to which they are immediately subordinate. Both objects may intersect partially or fully without restriction. The kind of intersectioning is ruled by the attribute "transparency". Its permissible values are 'transparent' and 'opaque'. Two cases may occur when a block is imaged :
- The block is 'opaque'; then any content of underlying objects is not imaged in the area of intersection.
- The block is 'transparent'; then the content of this block and any underlying content are combined.

When the structural part of a document has been analysed according to the above mentioned procedure the next imaging step is performed and the content imaging modules are called. The imaging of content is described by "Content Architectures", which are defined in ISO 8613/6–8. Three types of content are defined at present:
- text : ASCII characters
- raster graphics : pictorial information
- geometric graphics : geometric primitives

The character imaging module employs character bitmaps from a set of UNIX fonts. A probably better approach is the use of outline fonts, which reduces the amount of memory for font storage, but needs a fast processor for the generation

of character bitmaps. Constant spaced fonts and proportional fonts are supported by the current definition of the character content architecture. There is a limitation to only four character orientations, namely 0°, 90°, 180°, and 270°, which may be too restrictive.

Raster graphics, i.e. monochrome pictorial information with a single bit per pixel is processed by the raster imaging module, which provides for decompression of compressed data according to the CCITT Recommendation T.6 [4] and for scaling of raster images.

The most complex content to deal with is geometric graphics, which is encoded according to ISO 8632 [6] as Computer Graphics Metafile (CGM). The latter is a four–part standard. Part one contains a functional specification of a set of elements which may occur in a CGM file. Parts two to four describe three different encodings, which serve for different needs:

- the character encoding : minimum size of metafile
- the binary encoding : minimum effort to read and generate
- the clear text encoding: human readable form

An appropriate approach to rasterize geometric graphics is based on the Computer Graphics Interface (CGI) [7]. CGI is a graphics standard which offers functions for rendering graphical information on output devices, setting attributes, etc. Using this functionality, a CGM metafile can be interpreted and the corresponding CGI calls can be issued. For the development of such a transformation program the UNIX tool YACC is employed. The generation of such a transformation program, also referred to as CGM–Imager, is illustrated in figure 6.

Input for the YACC is a file consisting of three sections :

- declarations of the tokens and necessary data type definitions,
- the formal grammar of the CGM metafile (as listed in part one of ISO 8632), extended by actions that make the appropriate CGI calls,
- miscellaneous programs like a scanner routine, parameter decoding routines and an error routine.

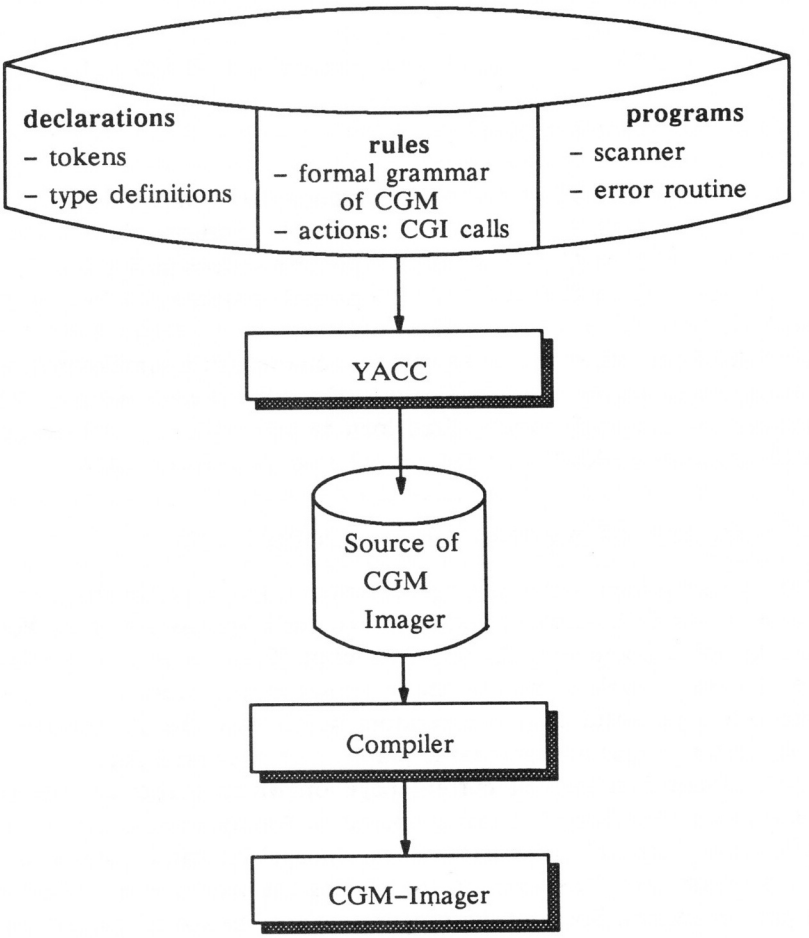
The YACC then generates the source of the aspired transformation program. After compiling it, any CGM metafile can be rendered on an output device which is equipped with a CGI interface. An appropriate manner to handle CGI–calls is the use of a hardware graphics processor that permits a fast transformation into the corresponding pixel data.

At the time of writing this paper the draft ODA standard does not contain provision for colour and grey–scale. It is, however, intended to include colour in all three existing content architectures in a similar manner.

We have considered part three of the CEPT Videotex standard T/CD 6–1 [8], which deals with photographic data, in order to derive a Videotex file for colour and grey–scale raster graphics. Such a file could be incorporated into an ODA document if colour or grey–scale images should be added.

The file format consists of a "Photographic Data Header Unit" and a "Photographic Data Transfer Unit". Both units are further subdivided into a Presentation Protocol Information (PPCI) part and a Presentation Protocol Data Unit (PPDU)

Fig. 6: Generation of the CGM–Imager



part. The PPCI part contains only control code, while the PPDU part of the header unit is a collection of all the relevant attributes and the PPDU part of the transfer unit contains the pictorial information according to the attributes specified in the header unit.

The attributes of the header PPDU are grouped into composition attributes and compression attributes. The following composition attributes are of special importance for hard copy rendition:

- COM : specifies the colour model
- RES : specifies the raster resolution
- BPC : specifies the bits per colour component

Currently, the CEPT standard defines only attributes suitable for the display of images on a screen. It is, however, quite simple to extend the present standard towards additional functionalities. The following example shows a possible extension of the COM attribute: (values in hex–notation)

<COM> := 2/0 3/C (definition according to T/CD 6–1)

3/C	Colour model	
3/0	YL U* V*	standardized in T/CD 6–1
3/1	R G B	possible extension
3/2	Y M C	possible extension
3/3	Y M C Black	possible extension

An extension of the other attributes can be achieved in a similar manner. Compression attributes permit the specification of a compression method. The compression methods presently standardized include techniques like Differential Pulse Code Modulation (DPCM) and Discrete Cosine Transform (DCT).

Further Approach to Render ODA Documents

Because of the growing importance of page description languages in the area of document printing, the ODA–Imager had been extended to generate also a PostScript description of a document. So far, PostScript [9,10] is in a very widespread use and already implemented by many printer manufacturers.

PostScript offers powerful graphic operators which ease the description of character text, raster images and geometric graphics on a printed page.

PostScript maintains an implicit current page on which marks are placed. These marks are set by PostScript's imaging operators. Among these operators are "fill" to mark areas, "stroke" to generate lines, "image" to paint raster images and "show" to create text. The location for placing the marks is described by PostScript's path operators. Some of these operators are "newpath" (initializes a new path), "moveto" (specifies the starting point), "rmoveto" (relative movement from the previous point), and "lineto" (connects the path with a straight line). Other PostScript operators allow to set attributes like font, line width, fill colour, etc.

The concept of outline fonts permits the choice of arbitrary character orientations and character sizes, limited only by the printer's raster resolution.

The transformation program from CDDS to PostScript is structured in the same way as the ODA–Imager. The modules for the processing of the hierarchical CDDS representation and the determination of the attribute values according to the "Defaulting Rules" are totally the same. Only the content imaging parts had to be replaced by modules which generate the appropriate PostScript commands and

write them to an external file. Finally, that file contains the page descriptions of the document and is sent to a PostScript printer where the hard copy is produced.

In this case the whole transformation is performed on the host system and not in the printer controller.

Conclusions

It has been shown how a document in ODIF form can be transformed into a bitmap for rendition on a hard copy device. One possibility is the rasterization of the ODIF document by the printer controller, which relieves the host system from a time consuming task.

A second approach with PostScript, a page description language, was discussed. Although the transformation from ODIF to PostScript must be performed by the host system, this approach has the advantage that there is variety of PostScript printers with high print quality already commercially available. PostScript is able to express all the features of an ODA document and has even capabilities beyond those of ODA, e.g, arbitrary character orientations and character sizes.

ODA still lacks further important features like colour or grey–scale content and cannot represent data types like tables and mathematical or chemical symbols. However, with respect to colour and grey–scale a preliminary concept is already in discussion. In this paper we have proposed the inclusion of colour and grey–scale raster graphics according to the CEPT Videotex Recommendation T/CD 6–1 for photographic content .

The work described was sponsored by the German Federal Ministry for Research and Technology (grant number ITM 8501 A) and the Commission of European Communities (ESPRIT project 295). Only the authors are responsible for the contents of this publication.

References

[1] The Seybold Report on Publishing Systems
Vol. 16, No. 2, September 22, 1986

[2] ISO DIS 8613, 1987-07-16, part 1-8
Information Processing - Text and Office Systems -
Office Document Architecture (ODA) and
Interchange Format

[3] ISO DIS 8613-5.2, 1987-07-16, part 5
Office Document Interchange Format (ODIF)

[4] CCITT Recommendation T. 6 : Facsimile Coding Schemes
and Coding Control Functions for Group 4 Facsimile
Apparatus

[5] P. Stucki : Advances in Digital Image Processing,
Plenum Press, New York, 1979, pp. 201-218

[6] ISO 8632 : Information Processing Systems - Computer
Graphics - Metafile for the Storage and Transfer of Picture
Description Information

[7] ISO TC97/SC21 N1179 (CGI Working Draft), Information
Processing, Computer Graphics, Interface Techniques for
Dialogues with Graphical Devices, May 1986

[8] CEPT Recommendation T/CD 6-1 :
Videotex Presentation Layer Data Syntax

[9] PostScript Language Reference Manual,
Addison-Wesley Publishing Company of Reading, Mass.

[10] PostScript Language Tutorial and Cookbook,
Addison-Wesley Publishing Company of Reading, Mass.